Amendments to the Drawings

The attached sheet of drawings includes changes to Fig. 16 to correct obvious typographical errors. This sheet replaces the original sheet of drawings.

Attachment: Replacement sheet.

Remarks

Applicants thank the Examiner for examining the claims of the present application. By this amendment, claims 1, 3, 5, 21, 24-25, 28, 31-32 and 38-40 are amended. Several of the claim amendments have been made for clarification purposes. Claim 41 is new. Upon entry of this amendment, claims 1-9 and 21-41 will remain pending. Applicants respectfully request reconsideration of the Examiner's rejections in view of the preceding amendments and following remarks.

Amended Independent Claims 1, 21 and 28 Comply with 35 U.S.C. § 112, 2nd paragraph

The Examiner rejects independent claims 1, 21 and 28 for allegedly failing to comply with 35 U.S.C. § 112, second paragraph. (Office action, pg. 2.) Applicants respectfully disagree with these rejections. However, Applicants have amended the claims as suggested by the Examiner to further the prosecution of the present application. Accordingly, the § 112 rejection of amended independent claims 1, 21 and 28 should be withdrawn and such action is respectfully requested.

Claims 1-9 and 21-40 are Patentable over a *De Armas/Ali* Combination

The Examiner rejects claims 1-9 and 21-40 as allegedly being obvious over U.S. Patent No. 5,864,819 ("*De Armas*") in view of *Ali et al.* ("Building Multi-Platform User Interfaces with UML," CADUI 2002). (Office action, pgs. 2-12.) This rejection is traversed.

Amended Independent Claim 1

Amended Independent claim 1 recites:

> A method of generating identifier data for persistently identifying a user interface element of interest in a graphical user interface of a source computer program, the method comprising:
>
> receiving data indicative of the user interface element of interest from a first software component; and
>
> in response to receiving the data indicative of the user interface element of interest, generating an element path identifier of the user interface element of interest for persistently and uniquely identifying the user interface element of interest across different states of the first software component and returning at least the unique element path identifier to the first software component;

wherein persistently and uniquely identifying the user element of interest comprises persistently and uniquely identifying the user interface element of interest across reboots of a computer running the source computer program.

Applicants respectfully submit that neither *De Armas* nor *Ali* (individually or in combination with one another) teach or suggest at least "generating an element path identifier of the user interface element of interest for persistently and uniquely identifying the user interface element of interest across different states of the first software component" as recited in amended independent claim 1.

For example, *Ali* describes UIML, a "language for building user interfaces for any device." (*Ali*, §1, second paragraph.) A UIML user interface definition contains "interface" elements comprising "all the UIML elements that describe [the user interface]." (*Ali*, § 4.1, second paragraph.) "Interface" elements are comprised of "structure" components that in turn are comprised of "parts" representing "the actual platform-specific UI Element[s]." (*Ali*, § 4.1, third paragraph.) *Ali* discloses transformations that convert generic UIML to platform-specific UIML, which can then be rendered using a UIML renderer. (*Ali*, Sec. 5.1, first paragraph.) Rendering can be accomplished "by compiling UIML into another language (e.g., WML or VoiceXML)." (*Ali*, § 2, third paragraph.)

Ali does not expressly disclose that UIML contains "part" identifiers. To the extent that Fig. 2 of *Ali* suggests that the "id" attributes in the UIML <part> tags (e.g., "id=TopPanel") act to identify "parts," such identifiers do not "uniquely identify[] the user element of interest across different states of the first software component" as in claim 1. Fig. 2 of *Ali* shows views of a user interface in multiple platforms (HTML and Java) transformed from a single UIML definition of the interface.

Further, *Ali* does not disclose that "element path identifier[s] of user interface elements ... [that] uniquely identify[] the user interface element ... across different states of the first software component" are generated either during the transformation of generic UIML code to platform-specific UIML code or in the rendering of platform-specific UIML code. The several types of transformations disclosed in § 5.1 of *Ali* disclose only the mapping of generic class names, part properties and events to platform-specific parts and the translation of behavioral element rules to platform-specific events. Accordingly, *Ali* does not teach or suggest "generating an element path identifier of the user interface element of interest for persistently and uniquely

identifying the user interface element of interest across different states of the first software component" as in claim 1.

Regarding *De Armas*, the Examiner admits that "*De Armas* does not specifically disclose identifying the user interface element of interest across different states of the first software component." (Office action, pg. 4.)

For at least the above reasons, the Examiner's 35 U.S.C. § 103(a) rejection of claim 1 based on *De Armas* in view of *Ali* should be withdrawn, and such action is respectfully requested.

Dependent Claims 2-9

The Examiner also rejects dependent claims 2-9 as allegedly being obvious over *De Armas* in view of *Ali*. (Office action, pgs. 5-7.) Claims 2-9 depend from amended claim 1, and are allowable for at least the reasons discussed above with respect to amended independent claim 1. Furthermore, claims 2-9 are allowable because of the new and nonobvious combinations of features recited in each respective claim.

Amended Independent Claim 21

Amended Independent claim 21 recites:

> At least one computer-readable medium having stored thereon computer-executable instructions related to a function responsive to a function call from a first software component, the function comprising:
>
> an input parameter representing a user interface element of interest in a graphical user interface of a source computer program;
>
> an output parameter representing an element path identifier for persistent unique identification of the user interface element of interest across multiple states of the source computer program, wherein the element path identifier comprises a hierarchical path of inheritance from the user interface element of interest to a parent root element; and
>
> executable software for receiving the input parameter representing a user interface element of interest and in response, generating the output parameter representing an element path identifier of the user interface element of interest such that the output parameter represents an identifier capable of persistently identifying the user interface element of interest across different builds of the source computer program.

Applicants respectfully submit that neither *De Armas* nor *Ali* (individually or in combination with one another) teach or suggest at least "an output parameter representing an element path identifier for persistent unique identification of the user interface element of interest across multiple states of the source computer program" as recited in amended independent claim 21.

For example, *Ali* describes UIML, a "language for building user interfaces for any device." (*Ali*, §1, second paragraph.) A UIML user interface definition contains "interface" elements comprising "all the UIML elements that describe [the user interface]." (*Ali*, § 4.1, second paragraph.) "Interface" elements are comprised of "structure" components that in turn are comprised of "parts" representing "the actual platform-specific UI Element[s]." (*Ali*, § 4.1, third paragraph.) *Ali* discloses transformations that convert generic UIML to platform-specific UIML, which can then be rendered using a UIML renderer. (*Ali*, Sec. 5.1, first paragraph.) Rendering can be accomplished "by compiling UIML into another language (e.g., WML or VoiceXML)." (*Ali*, § 2, third paragraph.)

Ali does not expressly disclose that UIML contains "part" identifiers. To the extent that Fig. 2 of *Ali* suggests that the "id" attributes in the UIML <part> tags (e.g., "id=TopPanel") act to identify "parts," such identifiers do not provide for "persistent unique identification of the user interface element of interest across multiple states of the source computer program" as in claim 21. Fig. 2 of *Ali* shows views of a user interface in multiple platforms (HTML and Java) transformed from a single UIML definition of the interface.

Further, *Ali* does not disclose that "output parameter[s] representing an element path identifier for persistent unique identification of ... user interface elements of interest across multiple states of the source computer program" are generated either during the transformation of generic UIML code to platform-specific UIML code or in the rendering of platform-specific UIML code. The several types of transformations disclosed in § 5.1 of *Ali* disclose only the mapping of generic class names, part properties and events to platform-specific parts and the translation of behavioral element rules to platform-specific events. Accordingly, *Ali* does not teach or suggest "an output parameter representing an element path identifier for persistent unique identification of the user interface element of interest across multiple states of the source computer program" as in claim 1.

Regarding *De Armas*, the Examiner admits that "*De Armas* does not specifically disclose identifying the user interface element of interest across different builds [and] across multiple states of the computer program." (Office action, pg. 8.)

For at least the above reasons, the Examiner's 35 U.S.C. § 103(a) rejection of claim 21 based on *De Armas* in view of *Ali* should be withdrawn, and such action is respectfully requested.

Dependent Claims 22-27

The Examiner also rejects dependent claims 22-27 as allegedly being obvious over *De Armas* in view of *Ali*. (Office action, pgs. 9-10.) Claims 22-27 depend from amended claim 21, and are allowable for at least the reasons discussed above with respect to amended independent claim 21. Furthermore, claims 22-27 are allowable because of the new and nonobvious combinations of features recited in each respective claim.

Amended Independent Claim 28

As understood by Applicants, *Ali* fails to remedy the deficiencies of *De Armas*. Therefore, in addition to separate reasons that Applicants will not belabor, claim 28 is allowable for at least the same reasons as claim 21.

Dependent Claims 29-34

The Examiner also rejects dependent claims 29-34 as allegedly being obvious over *De Armas* in view of *Ali*. (Office action, pgs. 10-11.) Claims 29-34 depend from amended claim 28, and are allowable for at least the reasons discussed above with respect to amended independent claim 28. Furthermore, claims 29-34 are allowable because of the new and nonobvious combinations of features recited in each respective claim.

Amended Independent Claim 35

As understood by Applicants, *Ali* fails to remedy the deficiencies of *De Armas*. Therefore, in addition to separate reasons that Applicants will not belabor, claim 35 is allowable for at least the same reasons as claim 21.

Dependent Claims 36-40

The Examiner also rejects dependent claims 36-40 as allegedly being obvious over *De Armas* in view of *Ali*. (Office action, pgs. 11-12.) Claims 36-40 depend from amended claim 35, and are allowable for at least the reasons discussed above with respect to amended independent claim 35. Furthermore, claims 36-40 are allowable because of the new and nonobvious combinations of features recited in each respective claim.

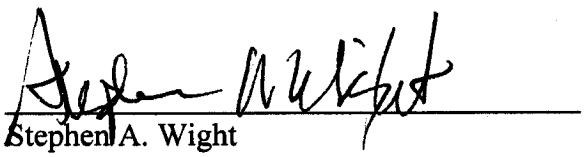
Conclusion

For the reasons cited above, the application is believed to be in condition for allowance and such action is respectfully requested. If any issues remain in light of these remarks and amendments, the Examiner is formally requested to contact the undersigned attorney to arrange for a telephonic interview. This request is being submitted under MPEP § 713.01, which indicates that an interview may be arranged in advance by a written request.

Respectfully submitted,

KLARQUIST SPARKMAN, LLP

One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Telephone: (503) 595-5300
Facsimile: (503) 595-5301

By 
Stephen A. Wight
Registration No. 37,759